NISTIR 6307

Initial Manufacturing Exchange Specification (IMES): Information Model for the Process Plan - Workstation Level

Y. Tina Lee

U.S. DEPARTMENT OF COMMERCE
Technology Administration
Manufacturing Systems Integration Division
National Institute of Standards and Technology
Manufacturing Engineering Laboratory
Gaithersburg, MD 20899

NIST

NISTIR 6307

Initial Manufacturing Exchange Specification (IMES): Information Model for the Process Plan - Workstation Level

Y. Tina Lee

U.S. DEPARTMENT OF COMMERCE
Technology Administration
Manufacturing Systems Integration Division
National Institute of Standards and Technology
Manufacturing Engineering Laboratory
Gaithersburg, MD 20899

March 1999

U.S. DEPARTMENT OF COMMERCE
William M. Daley, Secretary

TECHNOLOGY ADMINISTRATION
Gary R. Bachula, Acting Under Secretary for Technology

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
Raymond G. Kammer, Director

Initial Manufacturing Exchange Specification (IMES):

Information Model for the Process Plan – Workstation Level

Y. Tina Lee
Manufacturing Systems Integration Division
Manufacturing Engineering Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-0001

Preface

The Computer-Aided Manufacturing Engineering (CAME) program is developing a number of integrated software toolkits for design, manufacturing engineering, and production operations [1]. One of these toolkits is the Manufacturing Engineering ToolKit (METK) [2]. The goals of the METK project are 1) to develop generic interface specifications, data structures, information models, and data validation methodology for manufacturing engineering software; and, 2) to demonstrate an integrated toolkit using commercial, off-the-shelf software applications. The benefit will be an increase in the productivity of manufacturing engineers, and a reduction in the time required getting error-free, engineering data to the shop floor. The U.S. Navy Manufacturing Technology Program and the Systems Integration of Manufacturing Applications (SIMA) Program [3] at the National Institute of Standards and Technology (NIST) jointly fund the METK project.

The primary output of the METK project will be a collection of specifications called Initial Manufacturing Exchange Specifications (IMES) [4]. IMESs provide a means to meet the needs of the U.S. industry in the area of standards and testing methods by providing a structured approach to project activities in this arena. They will fill an important void in the manufacturing systems integration process as it exists today. Each IMES will be developed through an industry review and consensus process. It is expected that the manufacturing community will accept them as an authoritative specification for progressing or adopting as a national or international standard.

Three types of IMESs have been identified: an interface specification between a human being and a software application; an interface specification between two or more software applications; and a reference information repository specification. Each IMES involves components that define the integration aspect, specify a definitive solution to the integration problem, and demonstrate the validity of the proposed solution. It must contain a clear description of WHAT information the interface or repository MUST convey, and HOW it is conveyed. The content is usually specified by an information model of all the objects and related information attributes that are covered by the specification.

To support the scope and domain specifications, the IMES shall address a particular "example scenario," identifying an actual interface/information requirement derived from a real industrial problem. The proof of the value of the IMES to industry will be the ability to build a prototype to the IMES, using the software applications actually used by the industrial practitioners, and solving the cited problem. To support the

development of an IMES, projects will have seven phases: identify/define the industry need, conduct requirements analysis, develop proposed solution, validate proposed solution, build consensus, transfer technology, initiate standardization. Each of these phases has a well-defined set of deliverables.

The industry need for the METK was described in [5, 6]. An initial prototype for the METK was developed [2] in order to conduct the requirements phase of the IMES process (phase II). That prototype contained software applications to implement the following functions: product data management, workflow management, computer-aided design, generative process planning, and Numerical Control (NC) validation. The concept of a process plan was used to integrate the process planning and NC validation applications. This document follows the Phase II document of the requirement analysis for the process planning specification [7]. The Phase II document provides a scope and domain for the specification, a review of the related literature and standards, units of functionality, and the definition of required application objects. The process plan specification related work, including ISO 10303-213 or AP213 [9] and NIST Process Specification Language (PSL) [10], were briefed and compared in the Phase II document. In AP213, a process plan is viewed as a collection of the data needed to manufacture a part, however, in METK, a process plan includes not only the data but all instructions needed to manufacture a part [11]. The major differences between the METK project and the PSL project are representations of process (information model versus formal semantics,) scope (relative narrow scope versus wide range scope,) and project timeframe. The METK is addressing an integration problem that industry faces **today**, on the other hand, the PSL is addressing both industry's current integration challenges as well as the challenges they foresee in the future. Thus, a process plan IMES is needed in METK to support the exchange of the process plans between the METK applications. The domain for this process plan IMES is discrete parts manufacturing with emphasis on metal-removal machining operations.

This document, an IMES Phase III, presents an information model for a workstation-level process plan specification. The information model specifies the application objects, that were identified in the Phase II document [7], and the relationships between those application objects. The model is presented in the EXPRESS language [8] and contains the types, entity specifications, rules and functions. The information model is independent of any implementation method.

Certain commercial software and hardware products are identified in this paper. This does not imply approval or endorsement by NIST, nor does it imply that the identified products are necessarily the best available for the purpose.

1.0 INTRODUCTION

Hundreds of manufacturing engineering software applications have become available over the past decade. These applications can help manufacturing engineers perform the various tasks necessary to create manufacturing instructions from a product design. Industry forums hosted by NIST in Gaithersburg, MD, in 1995 [5, 6] identified integration of these engineering applications as a major unsolved problem. They indicated that a solution to this problem would result in significant timesaving in a product's life cycle.

The Manufacturing Engineering Toolkit (METK) Project was conceived to address this integration problem [2]. Its major goal is the identification of integration standards for implementing plug-compatible systems for manufacturing engineering functions. The METK Project is part of the Computer-Aided Manufacturing Engineering (CAME) Program in the Manufacturing Systems Integration Division (MSID) of Manufacturing Engineering Laboratory at NIST [1].

A prototype toolkit has been developed [2]. That prototype contains software applications to implement the following functions: product data management, computer-aided design, generative process planning, NC validation. Those applications include Matrix from Adra Systems, Pro-Engineer from Parametric Technology Corporation, ICEM Part from Control Data, VNC from Deneb Robotics. A major focus of the initial prototype was the integration of ICEM Part and VNC. This integration was achieved using the concept of a process plan [12, 13].

A requirements analysis for process planning integration in the METK was performed [7]. The process plan provides instructions on the sequence of processes for each product. This plan is for either a real, or a simulated, workstation – simulations are used to validate instructions before they are executed on the real system. The focus of the requirement analysis is a single workstation that contains a machine tool and its operators. As the result, a set of application objects used to integrate other planning and validation software applications was identified [7]. An information model based on these application objects is then documented in this report. The model is used for archiving, sharing, and exchanging the process planning information within the METK for workstation level plans. It is presented using the EXPRESS information modeling language [8].

The EXPRESS language has been developed as part of the ISO 10303 Standard for the Exchange of Product Model Data (STEP) activities [14]. STEP is an international standard, the result of an effort to develop a mechanism for digitally representing the physical and functional characteristics of a product throughout the product's life cycle. STEP includes definitions of information models and mechanisms for representing the models and related data. The information models provide a mechanism to communicate the structure and the semantics of the data necessary to exchange among different computer systems and environments.

Section 2 presents the METK process planning information model with the EXPRESS language and also with the EXPRESS-G form, a graphical notation for EXPRESS. Section 3.0 identifies some existing software tools that support the implementation of EXPRESS information models.

2.0 INFORMATION MODEL

In this section, an EXPRESS information model for representing a process plan within the METK is presented. An EXPRESS schema is composed of type declarations, entities, and constraints. The concept of a type in EXPRESS is similar to that of a data type in a standard programming language. It defines the kind of values that an object may assume. Entities are the main building blocks of an EXPRESS information model. An entity declaration describes the information content of an object, as well as the constraints on the object. Subsection 2.1 lists the types, entities and functions, and their definitions. The schema is presented in detail in subsection 2.2. Appendix A contains the listing of EXPRESS keywords that are used in the schema. The information model is also presented by EXPRESS-G, a graphical subset of EXPRESS. This EXPRESS-G information model is given in figures 2.3.1 through 2.3.7 of subsection 2.3.

2.1 Definitions for Types, Entities and Functions

This subsection specifies the objects, including types, entities, and functions, used in the information model. The objects and their definitions are given below.

2.1.1 Types

The following data types are used by the information model:

identifier – a string for an identification

month_type – an enumeration of the months of the year.

name – a string for naming an object.

resource_type – an enumeration of the types of the resources.

text – a string for a description

2.1.2 Entities

This subsection specifies all entities used by the information model. A brief description of each entity is given here, more detailed descriptions can be found in [7]. The following entities are used by the information model:

admin_information – a component of the pp_general_data entity, it consists of a general_info entity, a plan_revision entity, a plan_creation entity, and a process_planner entity.

attribute – a component of the work_element entity, it defines a characteristics of an activity within a process plan and consists of attribute's identification, type, description, and value.

cost_information – a component of pp_general_data, it consists of the estimator and estimated value of the cost for executing and completing the process plan.

customer_information – a component of pp_general_data, it consists of the identification and name of the organization for which the process plan is prepared and executed.

date – a calendar date, it consists of year, month and the day of the month.

facility_information – a component of the pp_general_data entity, it consists of the identification and description of the facility where the process plan is executed.

general_info – a component of an admin_information entity, it consists of process plan's identification, name, description, and notes.

part_information – a component of the pp_general_data entity, it consists of the identification and name of the part made using the process plan.

plan_creation – a component of the admin_information entity, it consists of process plan's creation date and creation time.

plan_revision – a component of the admin_information entity, it consists of process plan's revision date, revision time, and revision identification.

pp_general_data – a component of the process_plan_workstation_level entity, it is a collection of header information and is defined by an admin_information entity, a product_information entity, a part_information entity, a customer_information entity, a facility_information entity, a quality_information entity, and a cost_information entity.

pp_parameter – a component of the process_plan_workstation_level entity, it defines constant and variable parameters required for the execution of the process plan and consists of parameter's identification, name, and value, and description of the association with other entities.

pp_resource – a component of the process_plan_workstation_level entity, it defines the resources required to implement the process plan and consists of resource's type (controller, cutting tool, data file, employee, fixture, hand tool, machine, program, tool assembly, tool list, and work piece), name, identification, description, and quantity, and an identification for the file that contains additional pertinent information on the resources required for executing the process plan.

process_plan_workstation_level – information describing the process plan, it consists of a pp_general_data entity, a set of pp_resource entities, a set of pp_parameter entities, and a list of work_element entities.

process_planner – part of the admin_information entity, it consists of the identification and name of the person within an organization responsible for creating the process plan.

product_information – a component of the pp_general_data entity, it consists of the identification and name of the product for which the part is produced.

quality_information – a component of the pp_general_data entity, it consists of a quality coordinator and a quality level code.

time – the time of the date, it consists of hour, minute and second.

work_element – a component of the process_plan_workstation_level entity, it describes a specific action or activity required a procedure step and consists of the activity's description, value, resource and time estimation, a step identification, an immediate predecessor identifier, and a set of attribute entities.

2.1.3 Functions

A function is an algorithm that operates on parameters to produce a single resultant value of a specific data type. The following functions are used by the information model:

leap_year – a boolean function that checks whether a given integer could represent a leap year.

valid_date – a boolean function that checks a date for valid day, month, and year combinations.

valid_time – a boolean function that checks a time for valid minute and second combinations.

The entities include a broad range of data types, from simple entities to complex entities such as the work_element entity. The way these entity classes are related is specified by the model schema, which is described next.

2.2 Schema

This section describes the detailed information for the schema of process plan - workstation level. Types, entities, and functions are defined formally here.

```
SCHEMA process_plan_workstation_level_schema;

TYPE identifier = STRING;
END_TYPE;

TYPE month_type = ENUMERATION OF
    (April,
     August,
     December,
     February,
     January,
     July,
     June,
     March,
     May,
     November,
     October,
     September);
END_TYPE;

TYPE name = STRING;
END_TYPE;

TYPE resource_type = ENUMERATION OF
    (controller,
     cutting_tool,
     data_file,
     employee,
     fixture,
```

```
      machine,
      program,
      hand_tool,
      tool_assembly,
      tool_list,
      work_piece);
END_TYPE;

TYPE text = STRING;
END_TYPE;

ENTITY admin_information;
      planner : process_planner;
      revision : OPTIONAL plan_revision;
      creation : plan_creation;
      header_data : general_info;
END_ENTITY;

ENTITY attribute;
      attribute_id : OPTIONAL identifier;
      attribute_description : text;
      attribute_type : OPTIONAL text;
      attribute_value : REAL;
END_ENTITY;

ENTITY cost_information;
      cost_estimator : OPTIONAL identifier;
      cost_estimate_value : REAL;
END_ENTITY;

ENTITY customer_information;
      customer_id : OPTIONAL identifier;
      customer_name : name;
END_ENTITY;

ENTITY date;
      year : INTEGER;
      month : month_type;
      day : INTEGER;
 WHERE
      WR1: {1 <= day <= 31};
      WR2: valid_date (SELF);
      WR3: year > 0;
END_ENTITY;

ENTITY facility_information;
      facility_id : OPTIONAL identifier;
      facility_description : text;
END_ENTITY;

ENTITY general_info;
      plan_id : identifier;
      plan_description : OPTIONAL text;
      plan_name : OPTIONAL name;
      plan_notes : OPTIONAL text;
END_ENTITY;
```

```
ENTITY part_information;
     part_id : OPTIONAL identifier;
     part_name : name;
END_ENTITY;

ENTITY plan_creation;
     create_date : date;
     create_time : OPTIONAL time;
END_ENTITY;

ENTITY plan_revision;
     revision_id : OPTIONAL identifier;
     revision_date : date;
     revision_time : OPTIONAL time;
END_ENTITY;

ENTITY pp_general_data;
     customer_data : OPTIONAL customer_information;
     cost_data : OPTIONAL cost_information;
     admin_data : admin_information;
     product_data : OPTIONAL product_information;
     quality_data : OPTIONAL quality_information;
     facility_data : OPTIONAL facility_information;
     part_data : part_information;
 INVERSE
   Inv: process_plan_workstation_level FOR general_data;
END_ENTITY;

ENTITY pp_parameter;
     parameter_id : OPTIONAL identifier;
     parameter_name : name;
     parameter_association : OPTIONAL text;
     parameter_value : REAL;
END_ENTITY;

ENTITY pp_resource;
     resource_id : identifier;
     resource_description : OPTIONAL text;
     resource_type_item : resource_type;
     resource_file_id : OPTIONAL identifier;
     resource_quantity : OPTIONAL INTEGER;
     resource_name : OPTIONAL name;
END_ENTITY;

ENTITY process_plan_workstation_level;
     general_data : pp_general_data;
     procedure_step : LIST [1:?] OF work_element;
     resource_record : OPTIONAL SET [1:?] OF pp_resource;
     parameter_record : OPTIONAL SET [1:?] OF pp_parameter;
END_ENTITY;

ENTITY process_planner;
     planner_id : OPTIONAL identifier;
     planner_name : name;
END_ENTITY;
```

```
ENTITY product_information;
     product_id : OPTIONAL identifier;
     product_name : name;
END_ENTITY;

ENTITY quality_information;
     quality_coordinator_id : OPTIONAL identifier;
     quality_level_code : text;
END_ENTITY;

ENTITY time;
     hour : INTEGER;
     minute : OPTIONAL INTEGER;
     second : OPTIONAL INTEGER;
 WHERE
     WR1: valid_time (SELF);
END_ENTITY;

ENTITY work_element;
     attribute_data : OPTIONAL SET [1:?] OF attribute;
     activity_time_estimate : OPTIONAL REAL;
     activity_resource : text;
     activity_value : OPTIONAL text;
     activity_description : text;
     step_id : identifier;
     immediate_predecessor_id : identifier;
INVERSE
   Inv: process_plan_workstation_level FOR procedure_step;
END_ENTITY;

FUNCTION leap_year (year: INTEGER): BOOLEAN;
 IF ((((year MOD 4) = 0) AND ((year MOD 100) <> 0)) OR
    ((year MOD 400) = 0)) THEN
    RETURN (TRUE);
 ELSE
    RETURN (FALSE);
 END_IF;
END_FUNCTION;

FUNCTION valid_date (par: date): BOOLEAN;
 CASE par.month OF
   April: RETURN (par.day <= 30);
   June: RETURN (par.day <= 30);
   September: RETURN (par.day <= 30);
   November: RETURN (par.day <= 30);
   February : IF (leap_year (par.year)) THEN
                 RETURN (par.day <= 29);
              ELSE
                 RETURN (par.day <= 28);
              END_IF;
  OTHERWISE: RETURN (TRUE);
  END_CASE;
END_FUNCTION;

END_SCHEMA;
```

2.3 EXPRESS-G Figures

In this subsection, the information model is presented by EXPRESS-G, a graphical subset of EXPRESS. EXPRESS-G is represented by graphic symbols forming a diagram. The notation has three types of symbols: definition (symbols denoting data types and schema declarations), relationship (symbols describing relationships which exist among the definitions), and composition (symbols enabling a diagram to be displayed on more than one page). EXPRESS-G is defined in Annex D of ISO 10303-11:1994, EXPRESS Language Reference Manual [8]. The EXPRESS-G information model is given in figures 2.3.1 through 2.3.7.

Figure 2.3.1: Process_Plan_Operations_Sheet_Schema (EXPRESS-G: 1 of 7)

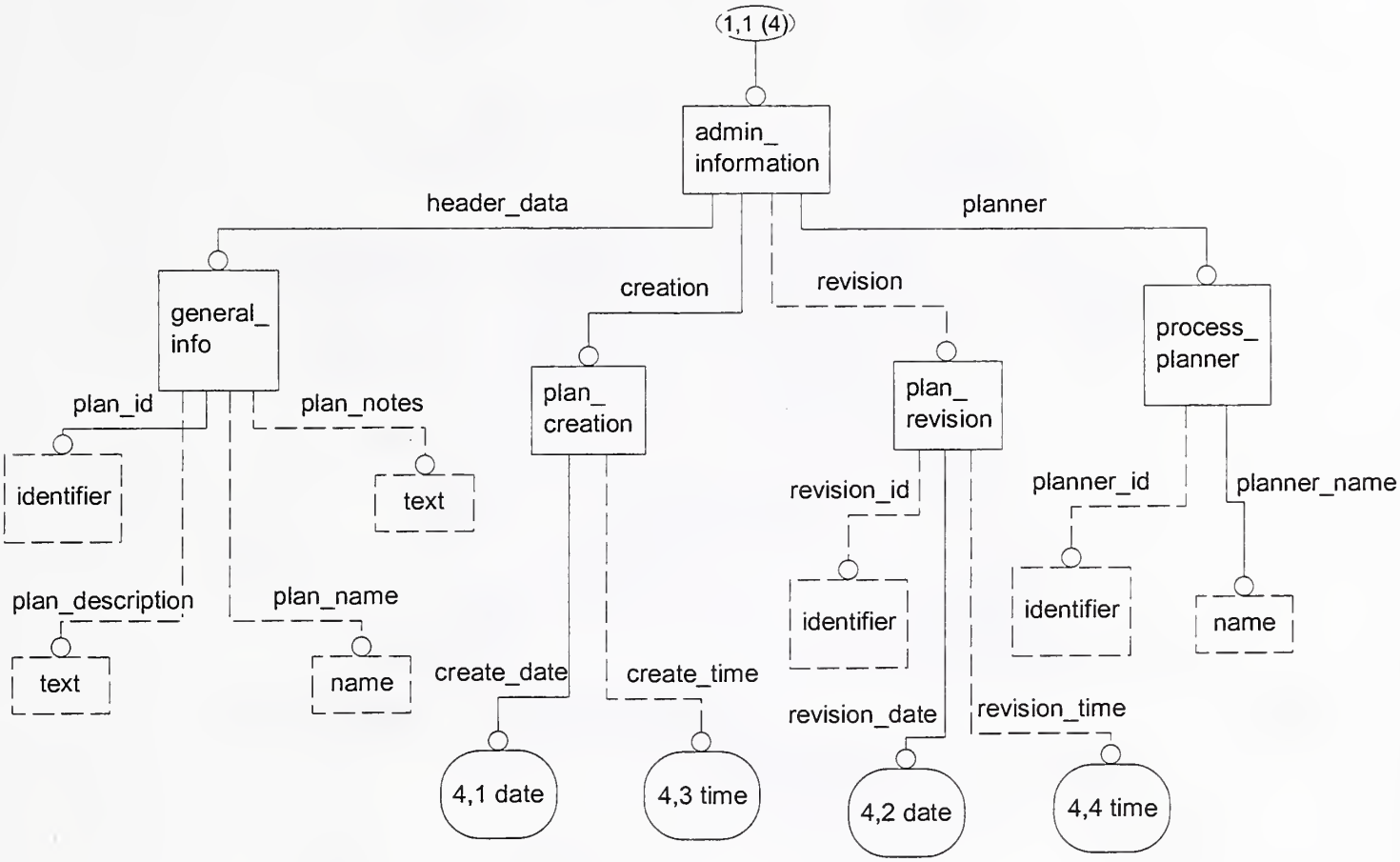


Figure 2.3.2: Process_Plan__Operations_Sheet_Schema (EXPRESS-G: 2 of 7)

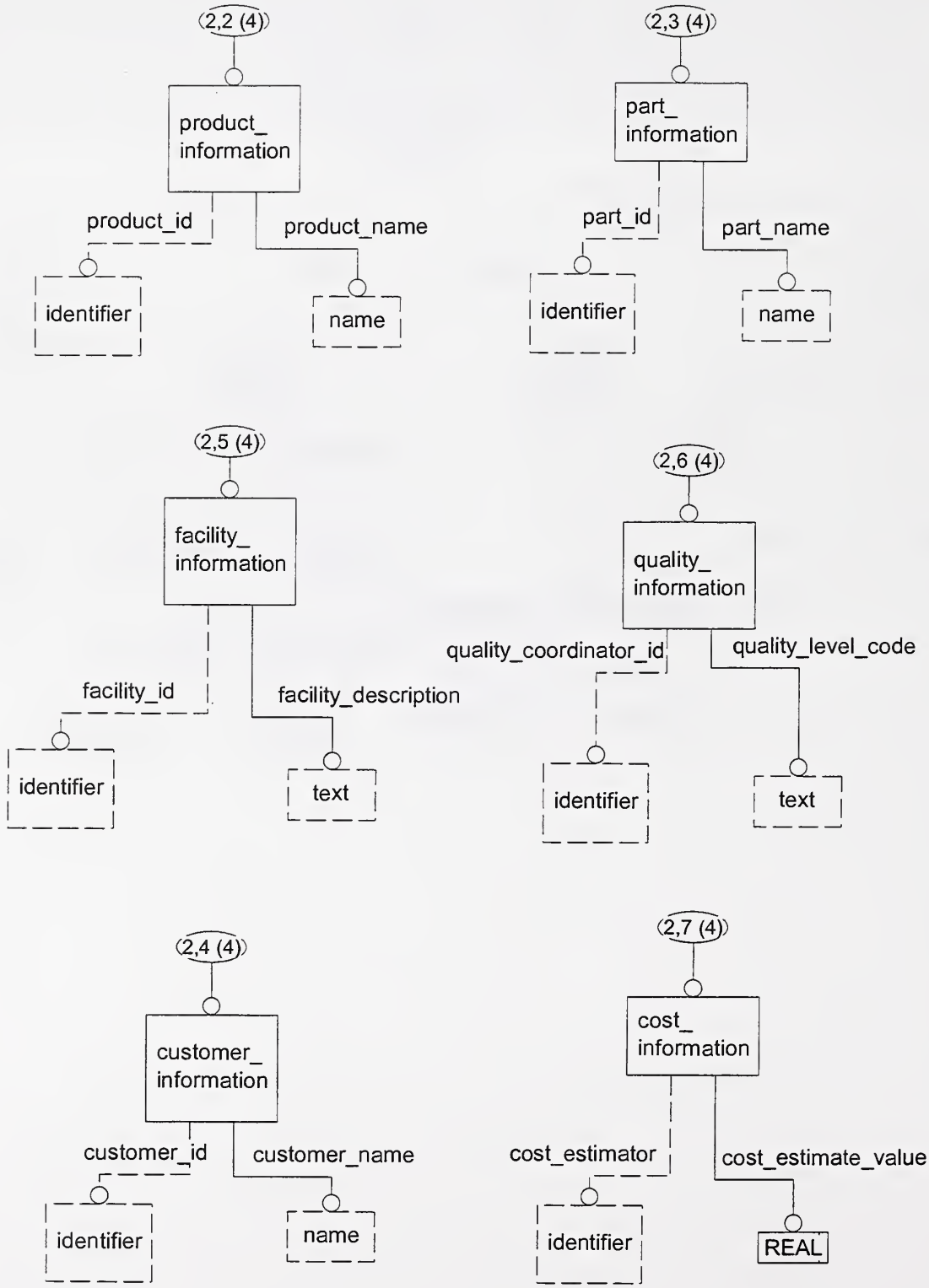


Figure 2.3.3: Process_Plan_Operations_Sheet_Schema (EXPRESS-G: 3 of 7)

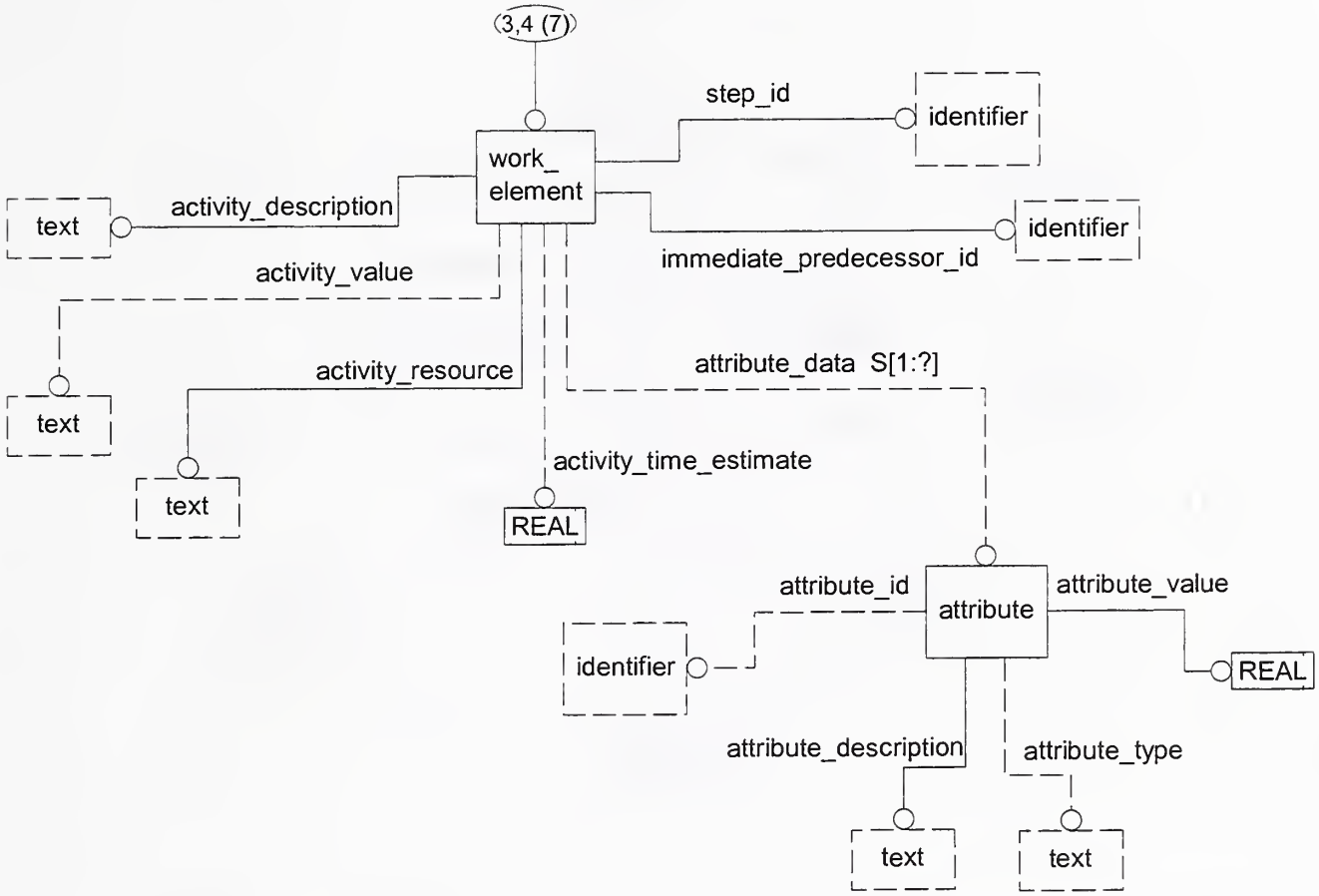
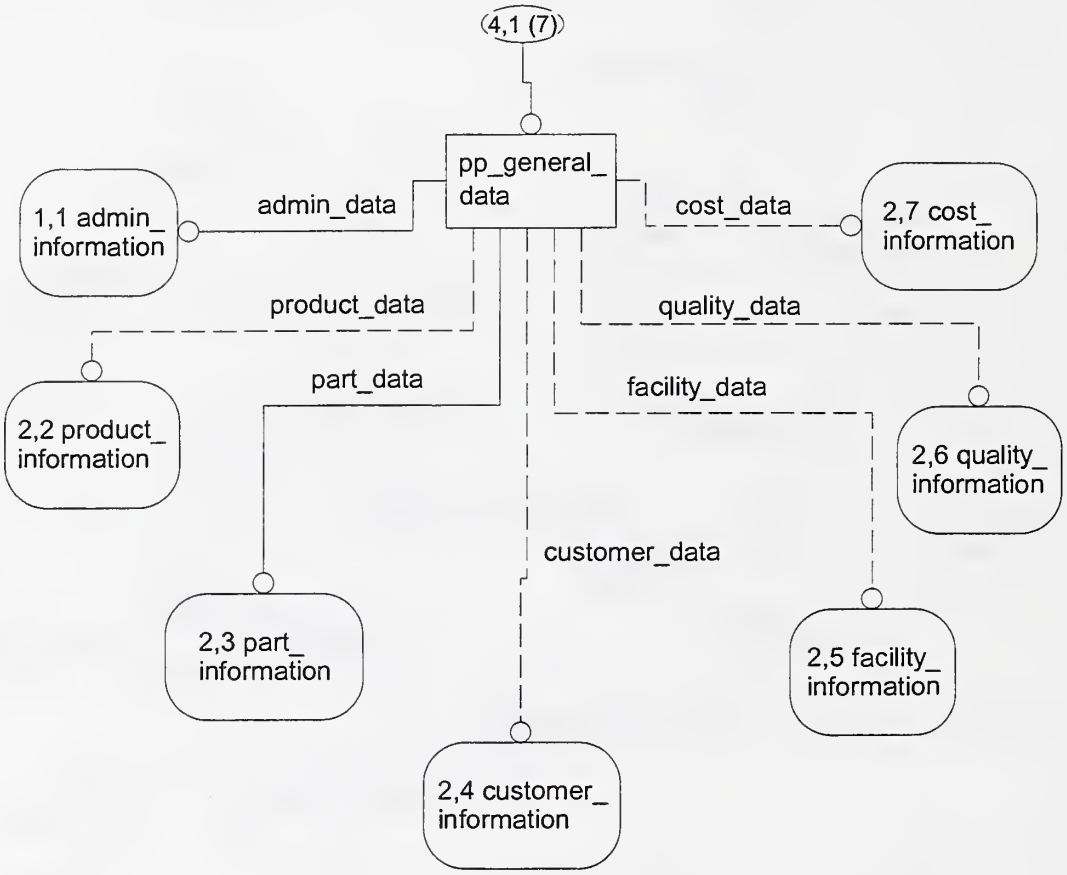
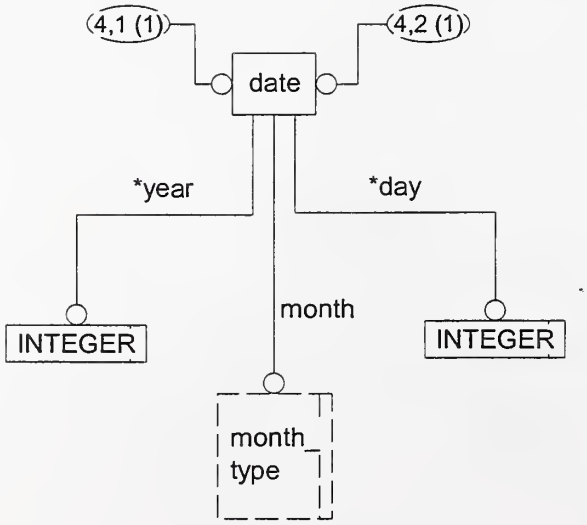


Figure 2.3.4: Process_Plan_Operations_Sheet_Schema (EXPRESS-G: 4 of 7)





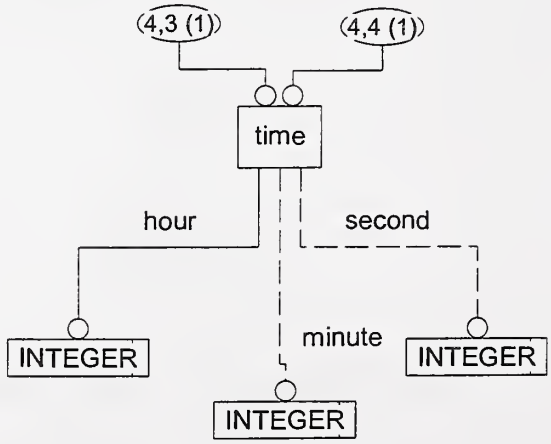


Figure 2.3.5: Process_Plan_Operations_Sheet_Schema (EXPRESS-G: 5 of 7)

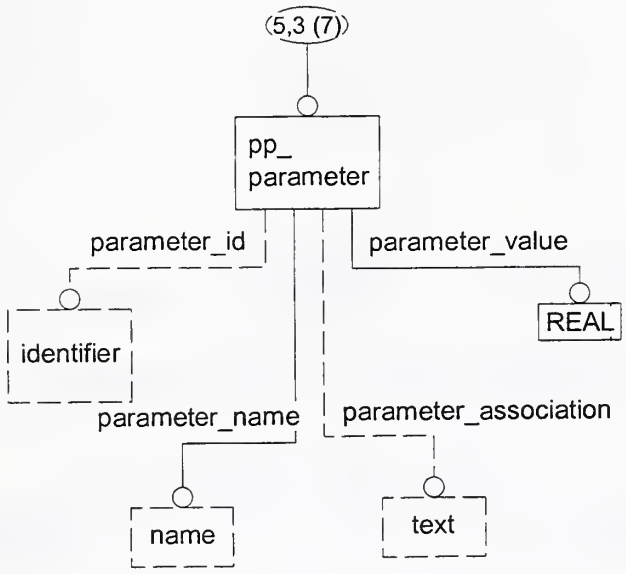


Figure 2.3.6: Process_Plan_Operations_Sheet_Schema (EXPRESS-G: 6 of 7)

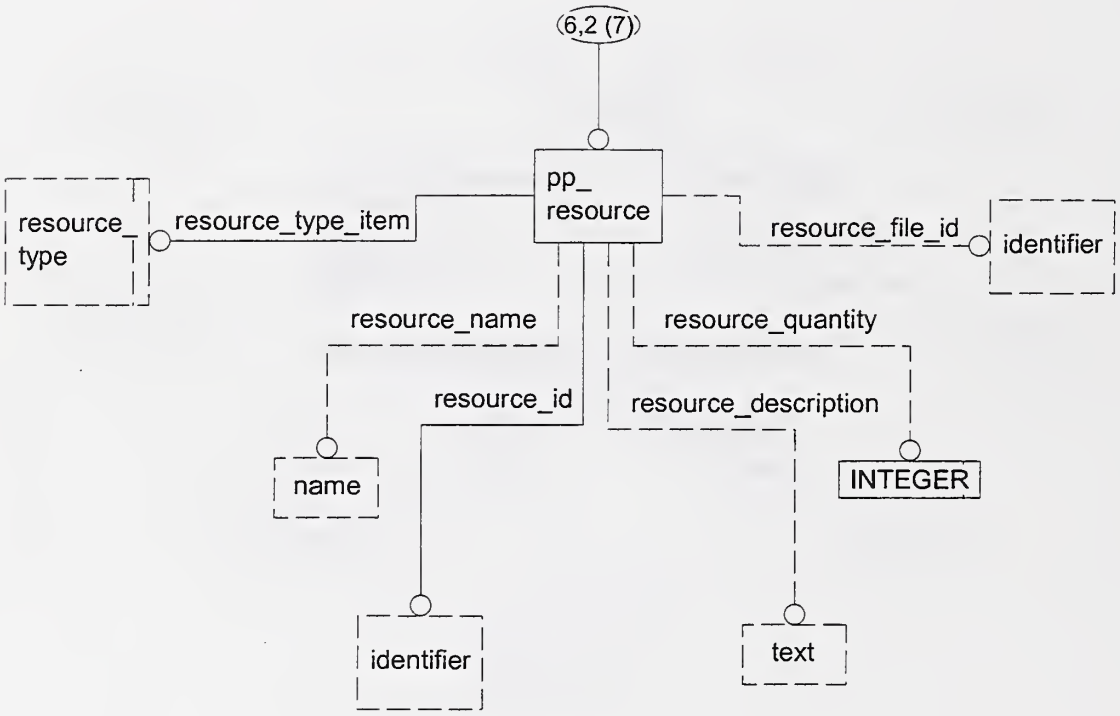
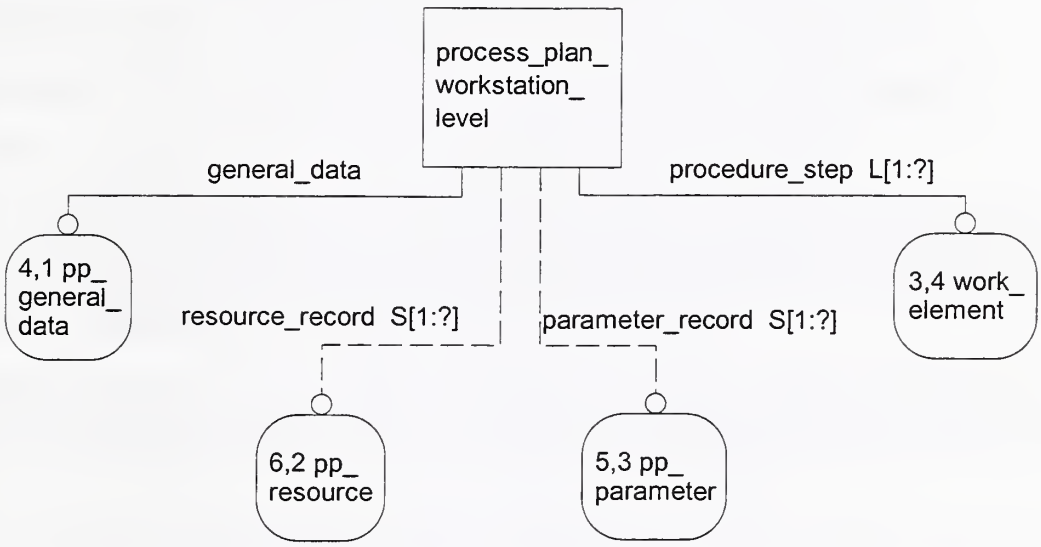


Figure 2.3.7: Process_Plan_Operations_Sheet_Schema (EXPRESS-G: 7 of 7)



3.0 USE OF THE INFORMATION MODEL

The information model presented in section 2.0 specifies the information necessary to represent process planning operations for the purpose of integrating other planning and validation tools within the METK environment and at the workstation level. The information model is independent of any implementation method. Several commercial and non-commercial software tools exist to support the implementation of EXPRESS information models. A document describing software tools and services for EXPRESS was published by Peter Wilson and is available from the STEP On-Line Information Service (SOLIS) [15, 16]. NIST has released a NIST STEP Toolset for manipulating STEP data; the Toolset is in the public domain and is also available from SOLIS. The implementors can take advantage of these software tools to generate various types of data structures from the information model in order to benefit the exchange of process plan data.

APPENDIX A:

The following is the list of EXPRESS keywords that are used in the process_plan_workstation_level_schema model. Brief definitions of these keywords are summarized for reader's convenience. Further information, refer to "The EXPRESS Language Reference Manual" [8].

AND - The reserve word AND is an AND operator. The AND operator requires two logical operands and evaluates to a logical value.

BOOLEAN - A BOOLEAN data type represents a TRUE or FALSE value.

CASE... OTHERWISE... statement – The CASE... OTHERWISE... statement is a mechanism for selectively executing statements based on the value of an expression.

END_ENTITY - The key word END_ENTITY is used to terminate an entity declaration.

END_SCHEMA - The key word END_SCHEMA is used to terminate a schema declaration.

END_TYPE - The key word END_TYPE is used to terminate a type declaration.

ENTITY - The key word ENTITY is used to specify an entity type. An entity type characterizes a collection of real-world physical or conceptual objects that have common properties. Any entity declared in a schema can be used as the data type of an attribute, local variable, or formal parameter. Using an entity as an attribute's data type establishes a relationship between the two entities.

ENUMERATION - The key word ENUMERATION is used to specify an enumeration data type. An enumeration data type is an ordered set of values represented by names. Each enumeration item belongs only to the data type that defines it and must be unique within that type definition.

FALSE – The reserve word FALSE is a LOGICAL constant representing the logical notion of falsehood. It is compatible with the BOOLEAN and LOGICAL data types.

FUNCTION - The key word FUNCTION is used to specify an algorithm which operates on parameters and produces a single resultant value of a specific data type.

IF... THEN... ELSE statement – The IF... THEN... ELSE statement allows the conditional execution of statements based on the expression of type LOGICAL.

INTEGER - The key word INTEGER is used to specify an integer data type. An integer data type represents a value of an integer number, the magnitude of which is unconstrained.

INVERSE - The key word INVERSE is used to constrain the relationship between two entities. If another entity has established a relationship with the current entity by way of an explicit attribute, an inverse attribute may be used to describe that relationship in the context of the current entity.

LIST - The key word LIST is used to specify a list data type. A list data type represents an ordered collections of like elements. The number of elements that can be held in a list can optionally be specified. If the size is not specified, the list can hold any number of elements. Duplicate elements are allowed in a list.

MOD – The reserve word MOD is an arithmetic operator for modulo. The MOD operator requires two operands and produces an integer result.

OF - The key word OF is used together with other keywords such as BAG, LIST, SET, ENUMERATION, SUBTYPE, SUPERTYPE, etc.

OPTIONAL - The key word OPTIONAL is used to indicate that the attribute need not have a value in order for an instance of that entity to be valid. In a given entity instance, an attribute marked as optional may have no actual value, in which case the value is said to be null. The null value function (NVL), which returns either the input value or an alternate value in the case where the input has a null value, may be used when a null value is unacceptable.

OR - The reserve word OR is an OR operator. The OR operator requires two logical operands and evaluates to a logical value.

REAL - The key word REAL is used to specify a real data type. A real data type represents rational, irrational, and scientific real numbers. Rational and irrational numbers have infinite resolution and are exact. Scientific numbers represent values that are known only to a specified precision.

RETURN statement – The RETURN statement terminates the execution of a FUNCTION or PROCEDURE.

SCHEMA - The key word SCHEMA is used to specify a schema type. A schema declaration creates a new scope in which the following objects may be declared: constant, entity, function, procedure, rule, and type.

SELECT - The key word SET is used to specify a set data type. A set data type represents an unordered collections of like elements. The number of elements that can be held in a set can be specified optionally. If the size is not specified, the set can hold any number of elements. No two elements of a set can have the same value.

SELF - The reserved word SELF is a constant that is considered to be a value of the base type.

SET - The key word SET is used to specify a set data type. A set data type represents an unordered collections of like elements. The number of elements that can be held in a set can be specified optionally. If the size is not specified, the set can hold any number of elements. No two elements of a set can have the same value.

STRING - The key word STRING is used to specify a string data type. A string data type represents a sequence of zero or more characters.

TRUE – The reserve word TRUE is a LOGICAL constant representing the logical notion of truth. It is compatible with the BOOLEAN and LOGICAL data types.

TYPE - The key word TYPE is used to specify a defined data type. A defined data type is a user extension to the set of standard data types. A defined data type can be used as any other data type by referencing the name given to it.

UNIQUE - The key word UNIQUE is used to specify a unique rule. A unique rule specifies either a single attribute name or a list of two or more attribute names. A rule that specifies a single attribute name is a "simple uniqueness constraint", requiring that any value of that attribute is associated with only one instance of that entity type. A rule that specifies two or more attribute names is a "joint uniqueness constraint",

requiring that any set of values, one from each of the named attributes, is associated with only one instance of that entity type.

WHERE - The key word WHERE is used to specify domain rules. Domain rules constrain the values of individual attributes or combinations of attributes for every entity instance.

REFERENCES:

[1] McLean, C., "Computer-Aided Manufacturing System Engineering," Advances in Production Management Systems, I.A. Pappas and I.P Tatsiopoulos (Editors) for Elsevier Science Publishers B.V. (North Holland), 1994.

[2] Iuliano, M., "Overview of the Manufacturing Engineering Toolkit Prototype," National Institute of Standards and Technology, NISTIR 5730, October 1995.

[3] Barkmeyer, E., Hopp, T., Pratt, M., and Rinaudot, G., "Background Study -Requisite Elements, Rationale, and Technology Overview for the System Integration for Manufacturing Applications (SIMA) Program," National Institute of Standards and Technology, NISTIR 5662, September 1995.

[4] Kemmerer. S., and Fowler, J., "Initial Manufacturing Exchange Specification (IMES) Concept Document", National Institute of Standards and Technology, NISTIR 5978, February 1997.

[5] Smith, M., and Leong, S., "Computer Aided Manufacturing Engineering Forum: First Technical Proceedings," National Institute of Standards and Technology, NISTIR 5699, March 1995.

[6] Smith, M., and Leong, S., "Computer Aided Manufacturing Engineering Forum: Second Technical Proceedings," National Institute of Standards and Technology, NISTIR 5846, August 1996.

[7] Ellis, K., Jones, A., and Lee, T., "Requirements Analysis: Process Plan Specification – Workstation Level," National Institute of Standards and Technology, NISTIR 6172, NIST, Gaithersburg, MD, 1998.

[8] ISO 10303-11:1994, "Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 11: Description Methods: The EXPRESS Language Reference Manual."

[9] ISO/DIS 10303-213, "Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 213: Application Protocol: Numerical Control Process Plans for Machined Parts," International Organization for Standardization, 1 rue de Varambe, Case Posatale 56, CH-1211 Geneva, Switzerland, 1994.

[10] Catron, B., and Ray S., "ALPS: A Language for Process Specification," International Journal of Computer Integrated Manufacturing, Vol. 4, No. 2, pp. 105-113, 1991.

[11] Iuliano, M., Jones, A., and Feng, S., "Analysis of AP213 for Usage as a Process Plan Exchange Format," National Institute of Standards and Technology, NISTIR 5992, NIST, Gaithersburg, MD, March 1997.

[12] McLean, C., "Interface Concepts for Plug-Compatible Production Management Systems," Computers in Industry - IFIP WG 5.7: Information Flow in Automated Manufacturing Systems, North Holland, Amsterdam, Netherlands, pp. 307-317, 1987.

[13] Iuliano, M., and Jones, A., "Controlling Activities in a Virtual Manufacturing Cell," Proceedings of WSC'96 Conference, San Diego, CA, December 1996.

[14] ISO 10303-1:1994, "Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 1: Overview and Fundamental Principles."

[15] Wilson, P. R., "EXPRESS Tools and Services," EXPRESS User Group Committee,1994.

[16] Rinaudot, G. R., "The IGES/PDES Organization: STEP On-Line Information Service (SOLIS) ," National Institute of Standards and Technology, NISTIR 5511, NIST, Gaithersburg, MD, 1994.